# Instructions to Authors for the Preparation of Papers for the 17th Australian International Aerospace Congress

***Please select category below:***
Normal Paper ☒
Student Paper ☐
Young Engineer Paper ☐

# Leveraging Industry Automation Techniques for Enhanced UAS System of Systems

Troy S. Bruggemann[1] and Duncan Campbell[2]

[1,2] *School of Electrical Engineering and Computer Science, Queensland University of Technology, 2 George St, Brisbane, Queensland, 4000, Australia*

## Abstract

With civilian UAS fast becoming a ubiquitous technology, and the availability of UAV protocols such as Micro Air Vehicle Link (MAVLink), there is great benefit to be gained by leveraging against established industrial automation systems and protocols. MAVLink is a CAN-like packet which carries messages and parameters for waypoint management, control, imagery, with multi-system and multi-component support for air and ground vehicles. OLE for Process Control (OPC) is an open industrial automation interoperability data standard which is widely used to expose industrial processes, sensors and controllers to a broad range of data consumers (clients) such as human machine interfaces, supervisory control systems, databases, and data analytics. MAVLink's context, reach and data management capabilities can be greatly expanded and abstracted through developing an OPC/MAVLink interoperability module and casting it into the OPC domain. This paper presents a simple OPC/MAVLink bridge demonstrator architecture with some indicative performance results.

**Keywords:** UAV**,** supervisory control, industrial automation, OPC, MAVLink, multi-agent systems.

## Introduction

It can be expected that highly automated machines such as UAVs will be an integral component of Industrial Internet of Things which promises greater productivity through increased interoperability between computing devices, machines, people and objects [1,2]. For example, industrial plants, may have autonomous UAVs and ground vehicles at the ready to perform inspection, checks, or transport goods and people.

Collaborative autonomy with heterogeneous machines and humans is foreseen to be the way forward and provide greater performance and resilience than non-collaborative approaches. Paradigms that make ambitious assumptions about high levels of machine autonomy, will not likely address the interim needs for operating numerous low SWaP (Size, Weight and Power) machines.

In order to bridge the gap between long-established industrial standards, civilian Command and Control (C2) and the civilian UAV technologies, this paper contributes an architecture

enabling industrial automation standard OPC to interoperate with UAVs and other devices running UAV communication protocols such as MAVLink. A demonstrator industrial automation/UAV protocol bridge has been developed motivated by the growing need of casting multiple UAV data sources to multiple data consumers.

The resulting OPC/MAVLink interoperability module (bridge) is described along with the hybrid OPC/MAVLink system level capability audit and performance assessment. This is in consideration of the potential for adoption in certain civilian UAS applications, including multiple UAV, manned-unmanned teaming and UAV-ground operations. This bridge does not replace UAV or industrial automation protocols, rather it shows how two widely used protocols in the two different domains can be connected seamlessly.

## Overview of Industrial Automation Protocols and UAV Communication Protocols

Using UAVs with conventional network infrastructure for communication and control has been considered before, but not combined with industrial supervisory control and data acquisition architectures [3]. OPC Unified Architecture (OPC UA) is an industrial M2M communication protocol for interoperability developed by the OPC Foundation [4,5]. OPC has an industry standard IEC 62541. OPC is implemented in server/client subscriptions and allows many computing platforms to communicate with industrial hardware devices such as PLC's. The OPC server is a software program that converts the hardware communication protocol used by a PLC into the OPC protocol. The benefits of OPC are that it includes a number of client interfaces including man-machine interfaces through GUIs, as well as inherent database driven trending and analysis, and alarm handling. A compliant OPC UA system has options for data encryption and secure authentication.

MAVLink is the most popular UAV communication protocol used in open source UAV autopilots, and has been adopted in ground robotics as well (via the Robot Operating System package called Mavros) [6]. MAVLink is a lightweight, header-only message marshalling library for micro air vehicles. MAVLink supports heartbeats from 1 second up to 1minute duration, has only 8 bytes overhead per packet and supports automatic code generation for new user-defined MAVLink messages. MAVLink relies upon the ITUX.25 checksum for packet corruption detection. It can support up to 255 aircraft and works on many microcontrollers and operating systems.

## System Architecture

The system architecture diagram is given in Fig. 1. The main components are the UAV side (or generally, any MAVLink compatible device) which consists of the UAV(s), Ground Control Station(s) (GCS) running software such as APM Planner, and other complementing optional GCS such as MAVProxy. We developed a MAV-OPC software bridge which enables sending UAV data in real-time to the server which can then be accessed by clients.

The system described in this paper is able to work with any OPC UA compliant client. Ignition [7] is the chosen system in this instance. Ignition is an Integrated Software Platform for Supervisory Control and Data Acquisition (SCADA) systems released by Inductive Automation which is based on an SQL database-centric architecture. Ignition SCADA modules provide features such as: real-time status control, alarming, reporting, data acquisition, scripting, scheduling, manufacturing executing systems (MES), and mobile support. The Ignition platform includes the Ignition Gateway, the Designer, and runtime clients. The client side contains an extensive number of client connections. Any client can access UAV data in (near) real-time or historical UAV data.

There are many possible C2 configurations that could be explored with this architecture. For example, if the UAV side is autonomous in the sense of automated flight from take-off to landing, then the client side can function as a remote C2 centre. Alternatively, a UAV operator in the field may be supported by the remote C2 centre, or the client side may simply be an observer.

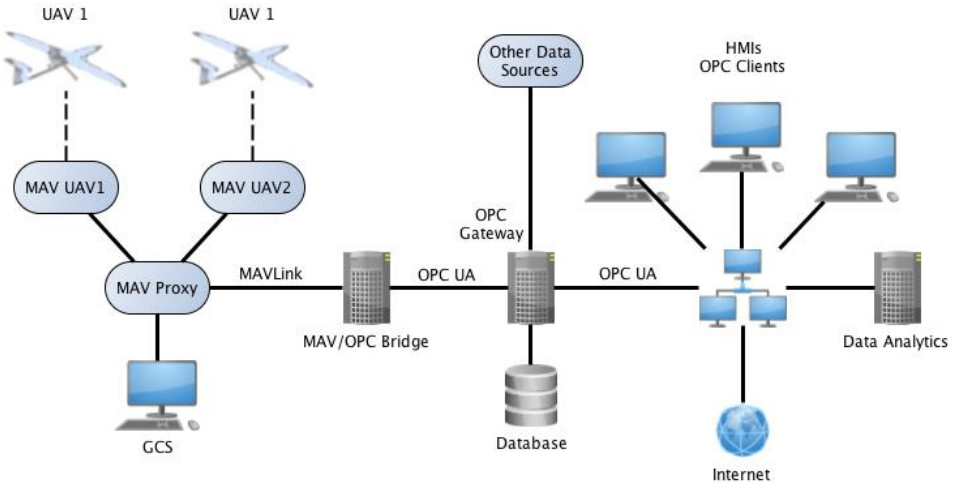


*Figure 1 System of Systems Architecture with OPC and MAVLink*

**OPC Unified Architecture Implementation**

We had previously verified that the approach of creating a bridge between OPC and MAVLink worked for OPC DCOM, and earlier implementation of OPC based on the Microsoft DCOM architecture. We then embarked on migrating the implementation to OPC Unified Architecture (OPC-UA) to allow for more robustness and open access. This section describes how the OPC-UA was implemented.

**OPC-UA Server, Ignition Gateway and Designer**

A OPC-UA Server solution was implemented with the Free OPC-UA Python library [8]. Then the Free OPC-UA server was connected to the Ignition Gateway (which has its own inbuilt web server) which allowed us to use all Ignition features. We adopted the Ignition Quick Client as the primary OPC client used during development and testing, for convenience. We then used the Ignition Designer which is a GUI for the Ignition software package. This allowed us to create and customize the visual layout, desired data tags, and visual features such as compass elements and level meters.

**MAV-OPC Software Bridge Description**

The MAV-OPC bridge is a piece of software coded in Python that enables communication between a MAV device and OPC-UA server. Its main job is to open a connection to the server, request data tag subscriptions and then parse data to/from the MAVLink data format (possibly going/coming to/from many UAVs) into the format required by OPC.

For the purpose of validating the bridge, we chose to implement the following MAVLink messages; HEARTBEAT which is a standard heartbeat message to maintain the connectivity between the ground station and machine; ATTITUDE which describes the UAV's attitude and attitude rate angles; GLOBAL_POSITION_INT which describes the UAV's position, velocities and heading; and VFR_HUD which describes the airspeed, groundspeed, heading, throttle, altitude and climb rate, which is information normally displayed on a UAV GCS. A full description of standard MAVLink messages is found at [6].

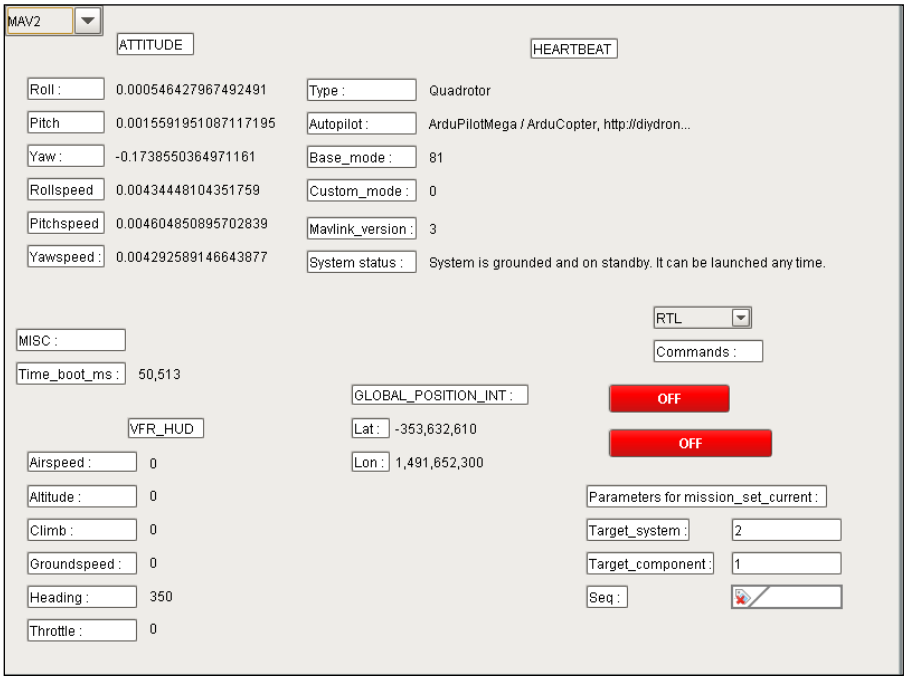


*Figure 2 The ignition GUI design for our development and debug purposes*

**Client GUI Design**

Using the Ignition Designer, we designed a basic functional GUI for our development and debugging purposes (Fig. 2). The GUI displays the information received for the four MAVLink messages we considered. We also included a drop-down textbox for selecting high level commands to send to the UAV (we chose RTL, or Return to Launch which commands the UAV to return to its launch site). It is also necessary to specify the intended recipient of the command (the target system) and (optionally) the target system's component (such as the flight controller or the onboard sensor payload).

OPC data tags are a single data-point (Fig. 4). The meaning of the tag depends upon the context. A tag for real-time data access provides real-time access to a value with a timestamp and quality indication. A tag can also be for historical data access that enables functionality such as trending, and can represent alarms and conditions.

## Results

Most of our development work was done in conjunction with the Ardupilot Software in the Loop (SITL) framework which runs the Ardupilot autopilot software and accurately simulates UAV flight and communications [9]. We tested our developed OPC server and MAV-OPC bridge software by simulating 4 UAVs flying in a circuit for a couple of hours and transmitting 1Hz MAVLink data over the local network infrastructure at Queensland University of Technology (QUT). We used the Ignition Designer to plot the real-time altitude, heading, latitude and longitude data for the 4 UAVs being logged onto QUT's OPC Gateway server database (Fig. 3).

Checking the performance statistics of the Gateway Server, we found that there was no noticeable delay and the CPU load due to our UAVs was less than 1% with <500MB RAM usage. At this time the 4 UAVs were running in addition to 4000 other processes, highlighting the feasibility of connecting UAVs with many other types of devices.

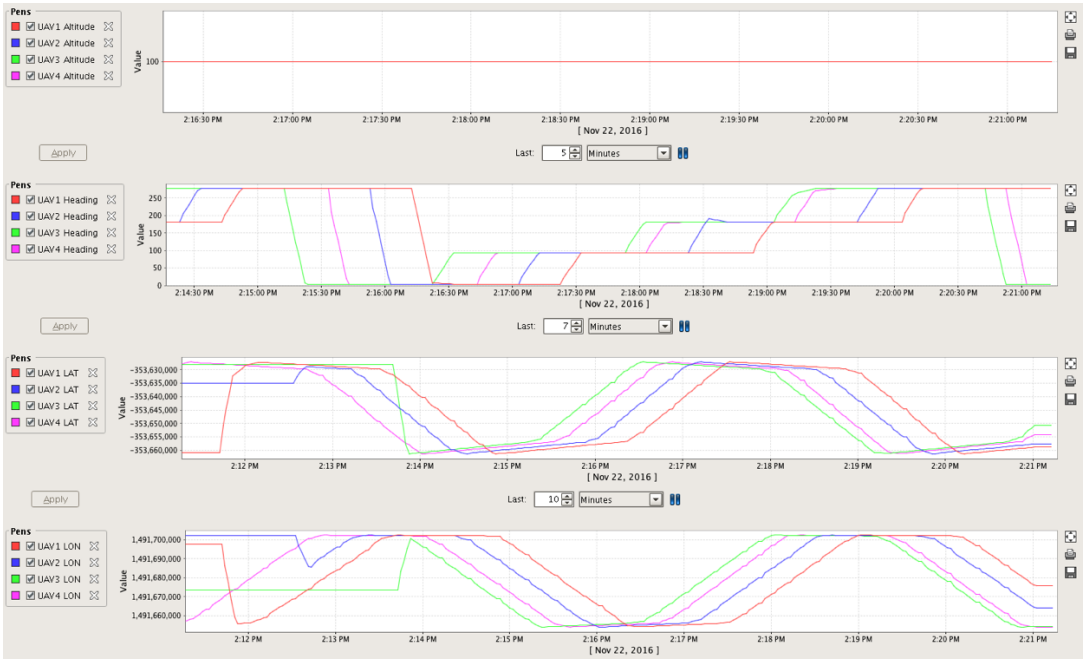


*Figure 3 Real-time and altitude, heading, latitude and longitude data for 4 simulated UAVs*

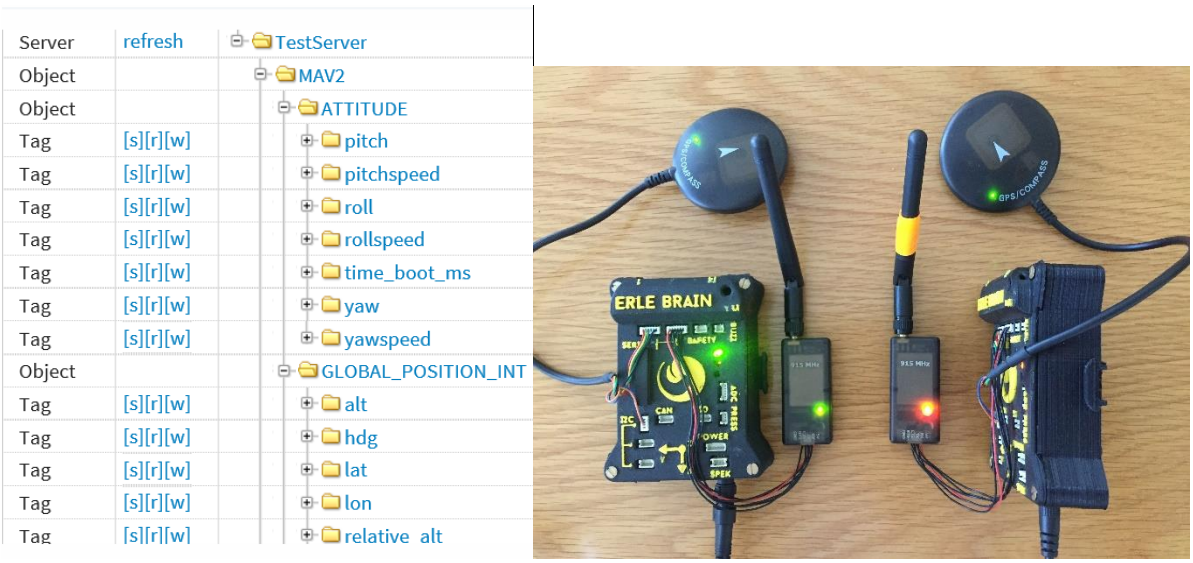


*Figure 2 Left: MAV tags in OPC client. Right: Two autopilots, one oriented at 90 degrees to test tag updating*

**Real UAV Sensor and State Data Reporting and High Level UAV Autopilot Command Test**

We then used hardware-in-the-loop testing with real autopilots to test the ability to report sensor and state data from multiple UAV autopilots to a centralized OPC server test bed. We also tested the ability to provide high-level UAV commands over OPC.

Firstly, two Erle Brain autopilots were started, with one purposely oriented at 90 degrees to the other to verify OPC tag updating (Fig. 4). Then the OPC server was run, and MAVLink data was streamed from both UAVs to the server via 900 MHz radio links. It was verified that data was recorded to the OPC-UA server by visualizing it on the Client GUI (Fig. 2).

To verify sending high level UAV commands over OPC to the UAVs, a value was written to an OPC tag to change the autopilot mode to ACRO (acrobatic) mode. The mode of the

autopilot was observed to change on the GCS, proving that the autopilot had changed mode. This test confirmed that connecting and sending commands to a real autopilot works.

## Conclusion

There is room to expand the development of the industrial automation/UAV protocol, GUI and interface design for advanced data analytics, visualization and decision making, and demonstrate its effectiveness in various applications. To be addressed is the lack of security protocols with free OPC. One further area of exploration could be the utilization of the data management and historical trending abilities of OPC which could be useful for assessing the operation and performance of UAVs for safety assessment and regulatory purposes.

## Acknowledgements

The authors thank M. Stapleton, E. Bellitto, T. Janvier and D. Bratanov for their contributions. This research was supported under the Department of Science, Information Technology and Innovation, Queensland State Governments Accelerate Fellowships Program.

## References

1. R. L. Shell and E. L. Hall, "The Future of Manufacturing," in Handbook of Industrial Automation, New York, CRC Press, 2000, pp. 451-452.

2. Brody, Paul, and Veena Pureswaran. "The next digital gold rush: how the internet of things will create liquid, transparent markets." Strategy & Leadership 43.1 (2015): 36-41.

3. M. Coombes, W. Eaton, O. McAree and W.-H. Chen, "Development of a Generic Network Enabled Autonomous Vehicle System," in UKACC International Conference on Control, Loughborough, 2014.

4. W. Mahnke, S.-H. Leitner and M. Damm, OPC Unified Architecture, Berlin: Springer Verlag Berlin Heidelberg, 2009.

5. OPC Foundation, "Home Page - OPC Foundation," OPC Foundation, 2015. [Online]. Available: https://opcfoundation.org/. [Accessed 19 August 2015].

6. L. Meier, J. Camacho, B. Godbolt, J. Goppert, L. Heng and M. Lizarraga, "MAVLink: Micro Air Vehicle Communication Protocol," 2011. [Online]. Available: http://qgroundcontrol.org/mavlink/start. [Accessed 18 August 2015].

7. Inductive Automation, "Scada Software - Ignition by InductiveAutomation.com," Inductive Automation, 2015. [Online]. Available: https://inductiveautomation.com/scada-software/ [Accessed 19 August 2015].

8. Free OPC-UA Library [Online]. https://github.com/FreeOpcUa/python-opcua [Accessed 19 August 2015].

9. 3DRobotics, "SITL Simulator (Software in the Loop) | Developer," 3DRobotics, February 2013. [Online]. Available: http:// ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html /. [Accessed 23 October 2015].